Mathematical Programming 21 (1981) 19–46.
North-Holland Publishing Company

A RESTRICTED LAGRANGEAN APPROACH TO THE TRAVELING SALESMAN PROBLEM

Egon BALAS*
Carnegie-Mellon University, Pittsburgh, PA., U.S.A.

Nicos CHRISTOFIDES

Imperial College of Science and Technology, London, England

Received 6 August 1979
Revised manuscript received 10 September 1980

We describe an algorithm for the asymmetric traveling salesman problem (TSP) using a new, restricted Lagrangean relaxation based on the assignment problem (AP). The Lagrange multipliers are constrained so as to guarantee the continued optimality of the initial AP solution, thus eliminating the need for repeatedly solving AP in the process of computing multipliers. We give several polynomially bounded procedures for generating valid inequalities and taking them into the Lagrangean function with a positive multiplier without violating the constraints, so as to strengthen the current lower bound. Upper bounds are generated by a fast tour-building heuristic. When the bound-strengthening techniques are exhausted without matching the upper with the lower bound, we branch by using two different rules, according to the situation: the usual subtour breaking disjunction, and a new disjunction based on conditional bounds. We discuss computational experience on 120 randomly generated asymmetric TSP's with up to 325 cities, the maximum time used for any single problem being 82 seconds. This is a considerable improvement upon earlier methods. Though the algorithm discussed here is for the asymmetric TSP, the approach can be adapted to the symmetric TSP by using the 2-matching problem instead of AP.

Key words: Traveling Salesman Problem, Assignment Problem, Branch and Bound, Lagrangean Relaxation, Hamiltonian Circuits, Arc Premiums/Penalties.

1. Outline of the approach

The traveling salesman problem (TSP), i.e., the problem of finding a minimum-cost tour (or hamiltonian circuit) in a directed graph $G = (N, A)$, can be formulated as the problem of minimizing

$$\sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}, \tag{1}$$

subject to

$$\left.\begin{aligned} \sum_{j \in N} x_{ij} &= 1, \quad i \in N, \\ \sum_{i \in N} x_{ij} &= 1, \quad j \in N, \end{aligned}\right\} \tag{2}$$

* Research supported by the National Science Foundation through grant no. MCS76–12026 A02 and the U.S. Office of Naval Research through contract no. N0014–75–C–0621 NR 047–048.

$$x_{ij} \in \{0, 1\}, \quad i, j \in N, \tag{3}$$

$$x \text{ is a tour.} \tag{4}$$

For $(i, j) \in A$, c_{ij} is the cost associated with the arc (i, j); for $(i, j) \notin A$, $c_{ij} = \infty$.

Condition (4) can be replaced (see [10]) by the subtour elimination inequaliti

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1, \qquad S \subset N, \quad 1 < |S| < n, \tag{5}$$

which can also be written as

$$\sum_{i \in S} \sum_{j \in N \setminus S} x_{ij} \geq 1, \qquad S \subset N, \quad 1 < |S| < n, \tag{6}$$

with $n = |N|$.

If $c_{ij} = c_{ji}$, $i, j \in N$, the TSP is *symmetric*; otherwise it is *asymmetric*. The symmetric TSP can be formulated more concisely on an undirected graph. However, in this paper we are concerned with the asymmetric case only.

The assignment problem defined by (1), (2) and

$$x_{ij} \geq 0, \quad i, j \in N \tag{7}$$

will be denoted by AP.

The TSP is a classical combinatorial optimization problem with wide applicability to the modeling of many important real-world situations in the areas of scheduling, sequencing, routing, etc. For background material see the surveys [4, 3], and more recently, [5, 9 and 15].

The TSP is NP-complete. Most exact methods for its solutions are enumerative, and they differ primarily in the way they generate lower bounds. The main vehicle for obtaining lower bounds is usually some easily solvable relaxation of the TSP, like the assignment problem [11, 17, 3, 7, 18], or the shortest 1-tree problem [13, 14, 6, 12]. The approach discussed in this paper is also enumerative, with the assignment problem as the relaxation used to derive lower bounds. Within this common general framework, however, our approach has essential new features:

(1) Unlike in the earlier procedures, the assignment problem used to derive lower bounds is a Lagrangean problem obtained from AP by subtracting from the objective function positive or negative multiples of the inequalities (5), (6) or their combinations. This has the effect of applying premiums or penalties to certain arc sets, and produces tight lower bounds.

(2) Instead of maximizing the Lagrangean dual by some iterative method involving the repeated solution of assignment problems, we restrict the multipliers to values that keep the solution to AP optimal for the modified objective function, and we give polynomially bounded procedures for approximating the maximum of the resulting restricted Langrangean problem. This keeps the lower bounds computationally cheap.

(3) We apply a tour-building heuristic to certain subgraphs defined by the agrangean form. This provides tight upper bounds at a low cost.

(4) Finally, we use some new branching rules that exploit problem structure.

As a result of these features, the procedure discussed here requires substantially smaller search trees and shorter computing times to solve randomly enerated asymmetric TSP's, than earlier methods.

We first outline the procedure, then discuss its various components in detail.

For any problem P, we denote by v(P) the value of (an optimal solution to) P. f S and T are node sets, we denote $(S, T) = \{(i, j) \in A \mid i \in S, j \in T\}$. Also, we lenote by X the set defined by the constraints (2), (7) of AP.

If we write the subtour elimination constraints (5), (6) in the generic form

$$\sum_{i \in N} \sum_{j \in N} a_{ij}^t x_{ij} \geq a_0^t, \quad t \in T, \tag{8}$$

hen the Lagrangean problem obtained from TSP by taking into the objective unction the inequalities (8) is

$$\max_{w \geq 0} L(w) \tag{9}$$

where

$$L(w) = \min_{x \in X} \sum_{i \in N} \sum_{j \in N} \left(c_{ij} - \sum_{t \in T} w_t a_{ij}^t \right) x_{ij} + \sum_{t \in T} w_t a_0^t.$$

The value of (9) is a lower bound on v(TSP).

Now let $\bar{x}$ be an optimal solution to AP, and consider instead of (9) the *estricted* Lagrangean problem

$$\max_{w \in W} L(w) \tag{10}$$

where

$$W = \{w \in R^{|T|} \mid w \geq 0 \text{ and } (u, v, w) \in Z \text{ for some } u, v \in R^n\}$$

and

$$Z = \left\{ (u, v, w) \mid u_i + v_j + \sum_{t \in T} w_t a_{ij}^t \begin{cases} = c_{ij} \text{ if } \bar{x}_{ij} = 1 \\ \leq c_{ij} \text{ if } \bar{x}_{ij} = 0 \end{cases} \right\}.$$

In (10), w is restricted to values for which the minimum in $L(w)$ is attained for $\bar{x}$. At first this seems a complicating factor; in fact, however, it makes it considerably easier to approximate the value of (10) than it would be to do the same thing for (9). Since (10) is a lower bound on (9), it is also a lower bound on v(TSP). Further, we have

Proposition 1.

$$\max_{w \in W} L(w) = \max_{(u, v, w) \in Z} \sum_{i \in N} u_i + \sum_{j \in N} v_j + \sum_{t \in T} w_t a_0^t. \tag{11}$$

Proof. If $\bar{w} \in W$ maximizes $L(w)$, the dual of the linear program $L(\bar{w})$ attains its maximum for a vector $(\bar{u}, \bar{v})$ such that $(\bar{u}, \bar{v}, \bar{w}) \in Z$ and

$$\sum_{i \in N} \bar{u}_i + \sum_{j \in N} \bar{v}_j + \sum_{t \in T} w_t a_0^t = L(\bar{w}). \tag{12}$$

Conversely, if $(\bar{u}, \bar{v}, \bar{w}) \in Z$ solves the problem on the right hand side of (11) then $\bar{x}$ solves the linear program $L(\bar{w})$, and (12) holds.

The restricted Lagrangean problem (10) has two useful properties. First, for any $(u, v, w) \in Z$, one has from (11)

$$\sum_{i \in N} u_i + \sum_{j \in N} v_j + \sum_{t \in T} w_t a_0^t \leq v(\text{TSP}), \tag{13}$$

i.e., any such (u, v, w) provides a valid lower bound on v(TSP). Second, while $\hat{x}$ remains an optimal solution to the assignment problem with the modified objective function $L(w)$, the process of maximizing $L(w)$ tends to create new alternative optima. Whenever such an alternative optimum $\hat{x}$ for some $w = \hat{w}$ turns out to be a tour, it has the following property.

Proposition 2. *If $\hat{x}$ satisfies with equality all inequalities* (8) *for which $\hat{w}_t > 0$, $\hat{x}$ is an optimal tour.*

Proof. $\hat{x}$ and $(\hat{u}, \hat{v}, \hat{w})$ are feasible solutions to the linear program (1), (2), (7), (8) and its dual, respectively, and in addition

$$\hat{u}_i + \hat{v}_j + \sum_{t \in T} \hat{w}_t a_{ij}^t = c_{ij}$$

whenever $\hat{x}_{ij} > 0$. This, together with the condition of the proposition, means that $\hat{x}$ and $(\hat{u}, \hat{v}, \hat{w})$ satisfy the complementary slackness conditions. Thus $\hat{x}$ is an optimal solution to (1), (2), (7), (8), hence an optimal tour.

We start by solving AP. At an arbitrary node of the search tree, we solve the assignment problem in the free variables. Next we use several procedures for generating an increasing sequence of lower bounds on v(TSP), by successively identifying inequalities (8) that

(i) are not satisfied by the current solution $\bar{x}$ to AP, and

(ii) admit a positive multiplier w_t which, together with $w_1, \ldots, w_{t-1}$, satisfies $w \in W$.

At a given stage, the *admissible graph* $G_0 = (N, A_0)$ is the spanning subgraph of

$\mathcal{G}$ containing those and only those arcs with zero reduced cost, i.e.,

$$A_0 = \left\{(i, j) \in A \mid u_i + v_j + \sum_{t \in T} w_t a_{ij}^t = c_{ij}\right\}.$$

When no more inequalities (8) satisfying conditions (i) and (ii) can be found, the admissible graph G_0 is strongly connected. We then store the bound given by (13) and try to find a tour in G_0. If a tour is found which satisfies with equality all inequalities associated with positive multipliers, it is optimal for the given subproblem. If a tour is found which violates this condition for some inequalities, attempts are made at finding new inequalities which satisfy the condition and admit positive multipliers. If successful, these attempts strengthen the lower bound, and they may also eliminate the inequalities that are slack. In any case, the value of the tour (in the original costs c_{ij}) provides an upper bound on v(TSP), while (13) provides a lower bound for the current subproblem; and we branch. Finally, if no tour is found in G_0, we add arcs to G_0 in the order of increasing reduced costs until a tour is found in the resulting graph. The cost of this tour again provides an upper bound on v(TSP), and we branch.

The assignment problems are solved by the Hungarian method, and the same method is used to recalculate the reduced costs whenever some u_i and v_j have to be changed. The bounding procedures are polynomial-time algorithms, considerably more efficient (in terms of improvement obtained versus computational effort) than earlier approaches, as evidenced by the computational results of Section 5. Searching the admissible graph G_0 for a tour is accomplished by a specialized implicit enumeration procedure, with a cut-off rule. Finally, for branching we use two different rules, one which derives a disjunction from a conditional bound, and one which breaks up a subtour.

A preliminary version of our approach, with fewer and less sophisticated bounding procedures, was discussed in [2].

2. Bounding procedures

We use three types of inequalities (8), and we will denote by T_1, T_2 and T_3 the corresponding subsets of T. For $t \in T$, let $\emptyset \neq S_t \subseteq N$ and $\bar{S}_t = N \setminus S_t$. An arc set of the form $K_t = (S_t, \bar{S}_t)$ is called a (directed) *cutset*. The first two types of inequalities are, in our current notation,

$$\sum_{(i,j) \in K_t} x_{ij} \geq 1, \quad t \in T_1 \tag{8a}$$

and

$$-\sum_{i \in S_t} \sum_{j \in S_t} x_{ij} \geq 1 - |S_t|, \quad t \in T_2, \tag{8b}$$

corresponding to (6) and (5) respectively.

For a given set S_t, the subtour elimination constraints (8a) and (8b) are

equivalent; nevertheless, an inequality (8a) may admit a positive multiplier wher the corresponding inequality (8b) does not (without changing other multipliers) and vice-versa.

For any $k \in N$ and $S_t \subset N \setminus \{k\}$, $S_t \neq \emptyset$, the arc sets

$$K_t' = (S_t, \bar{S}_t \setminus \{k\}) \quad \text{and} \quad K_t'' = (\bar{S}_t \setminus \{k\}, S_t)$$

are (directed) cutsets in the subgraph $\langle N \setminus \{k\} \rangle$ of G induced by $N \setminus \{k\}$. The third type of inequality used in our procedure is

$$\sum_{(i,j) \in K_t' \cup K_t''} x_{ij} \geq 1, \quad t \in T_3. \tag{8c}$$

Every inequality (8c) is the sum of the inequalities

$$-\sum_{i \in S_t \cup \{k\}} \sum_{j \in S_t \cup \{k\}} x_{ij} \geq -|S_t|, \tag{8b}_1$$

$$-\sum_{i \in S_t} \sum_{j \in S_t} x_{ij} \geq 1 - |S_t|, \tag{8b}_2$$

and the equations

$$\sum_{j \in N} x_{ij} = 1, \quad i \in S_t,$$

$$\sum_{i \in N} x_{ij} = 1, \quad j \in S_t.$$

Again, though the inequalities (8c) are implied by (8a), (8b) and (2), they often admit positive multipliers when the corresponding inequalities (8a) and (8b) do not.

The components of w associated with the inequalities (8a), (8b), and (8c) will be denoted by λ, μ and ν respectively.

2.1. Bounding procedure 1

This procedure starts by searching for an inequality (8a) which satisfies conditions (i), (ii) of Section 1; i.e., is violated by $\bar{x}$ and can be assigned a positive multiplier without changing the reduced costs. These conditions are satisfied for the inequality (8a) defined by a cutset K_t, if and only if

$$K_t \cap A_0 = \emptyset, \tag{14}$$

where A_0 is the arc set of the admissible graph G_0.

To find K_t satisfying (14), we choose any node $i \in N$ and form its reachable set $R(i)$ in G_0. If $R(i) = N$, there is no cutset $(S, \bar{S})$ with $i \in S$ satisfying (14), so we choose another $j \in N$. If for some $i \in N$, $R(i) \neq N$, then $K_t = (S, \bar{S})$ satisfies (14) for $S = R(i)$, and

$$\lambda_t = \min_{(i,j) \in K_t} \bar{c}_{ij} \tag{15}$$

is the largest value that can be assigned to the corresponding multiplier without making some reduced costs negative. We thus set $\bar{c}_{ij} \leftarrow \bar{c}_{ij} - \lambda_t$, $(i, j) \in K_t$, i.e., we apply a premium of λ_t to each arc of the cutset K_t. As a result, the arcs for which the minimum in (15) is attained become admissible, and we add them to A_0.

Next, we extend the reachable set $R(i)$ of node i by using the new arcs of G_0 and either find $R(i) = N$, or locate another cutset K_t satisfying (14). If $R(i) = N$, again we choose another node. This procedure ends when $R(i) = N, \forall i \in N$. At that stage G_0 is strongly connected, and $K \cap A_0 \neq \emptyset$ for all cutsets $K = (S, \bar{S})$, $S \subset N$.

Proposition 3. *Bounding procedure* 1 *stops after generating at most* $\frac{1}{2}(h-1)(h+2)$ *cutsets, where h is the number of subtours in* $\bar{x}$.

Proof. Starting with node i_1 belonging to subtour S_i, every cutset adds to G_0 an arc which includes into $R(i_1)$ a new subtour. After generating at most $h-1$ cutsets, $R(i_1) = |N|$. Now starting with node i_2 belonging to subtour $S_2 \neq S_1$ and proceeding to find $R(i_2)$, again at most $h-1$ cutsets can be generated. However, since we now have $i_2 \in R(i_1)$ and $i_1 \in R(i_2)$, the number of strong components of the current graph G_0 is at most $h-1$. Thus, continuing to find $R(i_3)$ for some node i_3 belonging to a subtour S_3, $S_1 \neq S_3 \neq S_2$, at most $h-2$ cutsets can be generated, and since the vertices of S_1, S_2 and S_3 now form a strong component, the number of strong components in the current graph G_0 is at most $h-2$. Continuing in the same way, the number of cutsets generated by the procedure (until G_0 becomes strongly connected) is at most

$$(h-1)+(h-1)+(h-2)+(h-3)+\ldots+1 = \tfrac{1}{2}(h-1)(h+2).$$

Generating a cutset that satisfies (14) or showing that none exists requires $O(mn)$ steps, where $n = |N|$, $m = |A|$.

Since the optimal dual variables $\bar{u}_i$, $\bar{v}_j$ associated with $\bar{x}$ are not changed by this procedure, and since

$$\sum_{i \in N} \bar{u}_i + \sum_{j \in N} \bar{v}_j = c\bar{x} = v(AP),$$

if T_1 is the index set of the inequalities generated by bounding procedure 1, the lower bound obtained for the current subproblem is, from (13),

$$B_1 = v(AP) + \sum_{t \in T_1} \lambda_t. \tag{16}$$

2.2. Bounding procedure 2

This procedure starts by searching for an inequality (8b) which is violated by $\bar{x}$ and admits a positive penalty without changing any of the λ_t, $t \in T_1$. If $S_1, \ldots, S_h$

are the node sets of the h subtours of $\bar{x}$, every inequality (8b) defined by S_t, $t = 1, \ldots, h$, is violated by $\bar{x}$; but a positive penalty μ_t can be applied without violating the condition that $\bar{x}_{ij} = 1$ implies $\bar{c}_{ij} = 0$, only by changing the values of some u_i and v_j, and only if an additional condition is satisfied. This condition can best be expressed in terms of the assignment tableau used in conjunction with the Hungarian method. A *line* of the tableau is a row or a column, a *cell* of the tableau is the intersection of a row and a column. Cells correspond to arcs and are denoted the same way.

Let S_t be the node set of a subtour of $\bar{x}$, let

$$A_t = \{(i, j) \in A_0 \mid i, j \in S_t\}, \qquad A'_t = \{(i, j) \in A_t \mid \bar{x}_{ij} = 1\}.$$

Proposition 4. *A positive penalty can be applied to the arcs with both ends in S_t if and only if there exists a set C of lines such that*

(i) *every $(i, j) \in A'_t$ is covered by exactly one line in C,*

(ii) *every $(i, j) \in A_t \smallsetminus A'_t$ is covered by at most one line in C,*

(iii) *no $(i, j) \in A_0 \smallsetminus A_t$ is covered by any line in C.*

If such a set C exists, and it consists of row set I and column set J, then the maximum applicable penalty is

$$\mu_t = \min_{(i,j) \in M} \bar{c}_{ij}, \tag{17}$$

where

$$M = (I, J) \cup (I, \bar{S}_t) \cup (\bar{S}_t, J). \tag{18}$$

Proof. Sufficiency. Suppose there exists a line set C, consisting of row set I and column set J, satisfying conditions (i), (ii), (iii). Then adding an amount $\mu > 0$ to $\bar{c}_{ij}$ for all $(i, j) \in (S_t, S_t)$, as well as to all $\bar{u}_i$, $i \in I$, and all $\bar{v}_j$, $j \in J$, produces a set of reduced costs $\bar{c}'_{ij}$ such that $\bar{c}'_{ij} = 0$ for all $(i, j) \in A'_t$, since $C = I \cup J$ satisfies (i). Further, from property (ii) of the set C, $\bar{c}'_{ij} \geq 0$, $\forall (i, j) \in A_t \smallsetminus A'_t$; and from (iii), $\bar{c}'_{ij} = \bar{c}_{ij} = 0$, $\forall (i, j) \in A_0 \smallsetminus A_t$. Thus the only reduced costs that get diminished as a result of the above modification, are those associated with arcs $(i, j) \in A$ for which either (α) nothing is added to $\bar{c}_{ij}$ and μ is added to $\bar{u}_i$ or to $\bar{v}_j$; or (β) μ is added to $\bar{c}_{ij}$ and to both $\bar{u}_i$ and $\bar{v}_j$. The two sets of arcs for which (α) holds are $(I, \bar{S}_t)$ and $(\bar{S}_t, J)$; whereas the arc set for which (β) holds is (I, J). The union of these three arc sets is M defined by (18). Thus a positive penalty at most equal to μ_t defined by (17) can be applied to the arc set (S_t, S_t) in the above described manner without producing any negative reduced costs.

Necessity. Suppose a penalty $\mu > 0$ can be applied to the arc set (S_t, S_t). Since adding $\mu > 0$ to $\bar{c}_{ij}$, $\forall (i, j) \in (S_t, S_t)$, produces positive reduced costs for all $(i, j) \in A_t$, in order to obtain reduced costs $\bar{c}'_{ij} = 0$ for all $(i, j) \in A'_t$, one must increase by μ the sum $\bar{u}_i + \bar{v}_j$ for all $(i, j) \in A'_t$. It is easy to see that if this can be

done, then it can be done by adding μ to $\bar{u}_i$ or $\bar{v}_j$ (but not to both), for every $(i, j) \in A_t'$; hence there exists a set C of lines satisfying condition (i). Further, if $(i, j) \in A_t \smallsetminus A_t'$, then μ cannot be added to both $\bar{u}_i$ and $\bar{v}_j$ without creating $\bar{c}_{ij}' < 0$, hence C must satisfy (ii). Finally, if $(i, j) \in A_0 \smallsetminus A_t$, then μ cannot be added to either $\bar{u}_i$ or $\bar{v}_j$ without making $\bar{c}_{ij}' < 0$, hence condition (iii) must also hold.

Given the node set S_t of a subtour, we have to check whether a set of lines C satisfying (i), (ii), (iii) exists. This can be done as follows.

First, every row $i \in S_t$ such that $(i, j) \in A_0$ for some $j \in N \smallsetminus S_t$, and every column $j \in S_t$ such that $(i, j) \in A_0$ for some $i \in N \smallsetminus S_t$, can be ruled out as a candidate for entering C. Let R and K be the index sets of such rows and columns respectively, and let

$$I_1 = \{i \in N \mid (i, j) \in A_t' \text{ and } j \in K\}, \qquad J_1 = \{j \in N \mid (i, j) \in A_t' \text{ and } i \in R\}.$$

Since by (i) every cell of A_t' must be covered by at least one line in C, C must contain $I_1 \cup J_1$. For the same reason, if $A_t' \cap R \cap K \neq \emptyset$, then no positive penalty can be applied to the arc set (S_t, S_t).

Since by (i) and (ii) every cell of A_t must be covered by at most one line in C, if $A_t \cap I_1 \cap J_1 \neq \emptyset$, then again no positive penalty can be applied to the arc set (S_t, S_t). Now assume both sets are empty. Then if (I_1, J_1) covers A_t', we set $C = I_1 \cup J_1$ and we are done; otherwise we use the Hungarian algorithm to complete the search for a cover satisfying (i), (ii), (iii). If such a cover exists, the Hungarian algorithm finds it, and μ_t given by (17) can be applied as a penalty; otherwise the Hungarian method finds a cover which violates some of the conditions (i), (ii), (iii), in which case no positive penalty can be applied.

If T_2 is the index set of the inequalities (8b) which admit positive penalties μ_t, we have the following:

Proposition 5. *A lower bound on the value of the current subproblem is given by*

$$B_2 = B_1 + \sum_{t \in T_2} \mu_t. \tag{19}$$

Proof. Whenever a penalty $\mu_t > 0$ is applied to an arc set (S_t, S_t) associated with a constraint (8b), the cost function of AP is modified, and the value of the solution $\bar{x}$, hence also the value of a solution to the dual of $\overline{\text{AP}}$, the assignment problem with the modified costs, is increased by $|S_t|\mu_t$. Thus, after applying $|T_2|$ penalties μ_t, the value of an optimal solution $(\hat{u}, \hat{v})$ to the dual of $\overline{\text{AP}}$ is

$$\sum_{i \in N} \hat{u}_i + \sum_{j \in N} \hat{v}_j = v(\text{AP}) + \sum_{t \in T_2} |S_t|\mu_t.$$

Using (13), and noting that $a_0^t = 1$ for $t \in T_1$ and $a_0^t = 1 - |S_t|$ for $t \in T_2$, we

obtain the lower bound

$$B_2 = \sum_{i \in N} \hat{u}_i + \sum_{j \in N} \hat{v}_j + \sum_{t \in T_1} \lambda_t + \sum_{t \in T_2} (1 - |S_t|)\mu_t$$

$$= B_1 + \sum_{t \in T_2} \mu_t.$$

2.3. Bounding procedure 3

This procedure searches for inequalities of the form (8c) which are violated by $\bar{x}$ and admit a positive multiplier ν_t without requiring changes in the multipliers assigned earlier. This is done by checking for each node whether it is an articulation point of G_0. If node k is an articulation point, i.e., if the subgraph $\langle N - \{k\}\rangle$ of G_0 is disconnected, with S_t as one of its components, then denoting $K_t' = (S_t, \bar{S}_t \smallsetminus \{k\})$ and $K_t'' = (\bar{S}_t \smallsetminus \{k\}, S_t)$ we have

$$K_t' \cap A_0 = \emptyset, \qquad K_t'' \cap A_0 = \emptyset.$$

Thus we can apply a positive premium to the arcs in the pair of cutsets K_t', K_t'', whose value is

$$\nu_t = \min_{(i,j) \in K_t' \cup K_t''} \bar{c}_{ij}. \tag{20}$$

If T_3 is the index set of all those inequalities (8c) found to admit a positive multiplier, at the end of bounding procedure 3 we have (from (13) and (19)) the lower bound

$$B_3 = B_2 + \sum_{t \in T_3} \nu_t$$

$$= v(\mathrm{AP}) + \sum_{t \in T_1} \lambda_t + \sum_{t \in T_2} \mu_t + \sum_{t \in T_3} \nu_t. \tag{21}$$

If at any time during the bounding procedure the current lower bound matches (or exceeds) the upper bound given by the value of the best available tour, the current subproblem is fathomed and we turn to another node of the search tree. Otherwise, after obtaining the bound B_3 we try to find a tour in G_0.

2.4. Example

Consider the 9-city TSP whose cost matrix is shown in Table 1.

The solution to AP has value 31. The reduced cost matrix $[\bar{c}_{ij}]$ is shown in Table 2 and the solution $\bar{x}$ is given by $\bar{x}_{ij} = 1$ for those (i, j) corresponding to boxes in that matrix, $\bar{x}_{ij} = 0$ otherwise. The corresponding admissible graph is shown in Fig. 1.

Bounding procedure 1. Cutset $K_1 = (\{1, 2, 3\}, \{4, 5, 6, 7, 8, 9\})$ admits $\lambda_1 = 4$, and cutset $K_2 = (\{4, 5\}, \{1, 2, 3, 6, 7, 8, 9\})$ admits $\lambda_2 = 3$. The lower bound, from (16), becomes $B_1 = 31 + 4 + 3 = 38$. The new reduced cost matrix is shown in Table 3 and the corresponding admissible graph in Fig. 2.

Table 1

$[c_{ij}]$ =	1	2	3	4	5	6	7	8	9
1	×	2	8	11	15	12	12	11	13
2	8	×	4	12	18	14	12	14	17
3	6	9	×	15	20	17	13	10	17
4	13	15	17	×	5	8	11	15	16
5	10	14	16	3	×	16	12	15	13
6	7	7	11	9	14	×	3	7	8
7	5	7	6	2	4	11	×	2	8
8	4	10	7	9	12	10	13	×	4
9	9	5	9	11	8	2	7	7	×

Table 2

$[\bar{c}_{ij}]$ =	1	2	3	4	5	6	7	8	9
1	×	[0]	6	9	13	10	10	9	11
2	4	×	[0]	8	14	10	8	10	13
3	[0]	3	×	9	14	11	7	4	11
4	8	10	12	×	[0]	3	6	10	11
5	7	11	13	[0]	×	13	9	12	10
6	4	4	8	6	11	×	[0]	4	5
7	3	5	4	0	2	9	×	[0]	6
8	0	6	3	5	8	6	9	×	[0]
9	7	3	7	9	6	[0]	5	5	×

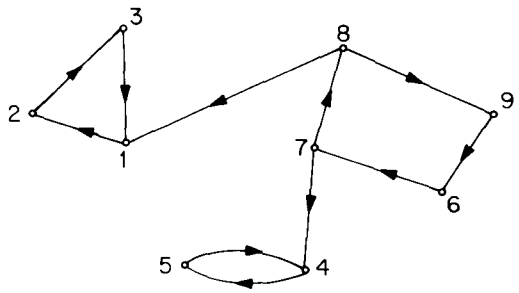

Fig. 1. Graph G_0 defined by the AP solution.

Bounding procedure 2. The subtours of the AP solution are (1, 2, 3), (4, 5) and (6, 7, 8, 9). Subtours (1, 2, 3) and (6, 7, 8, 9) do not admit positive values of μ_t. However, inequality (8b) for subtour (4,5) is

$$-(x_{45}+x_{54}) \geq -1,$$

and a set C of lines of the matrix of Table 3 satisfying the conditions of Proposition 7 for this subtour is given by: (row 5, column 5). From this set C we compute $\mu_t = 2$, and the lower bound becomes $B_2 = 38+2 = 40$.

Table 3

$[\bar{c}_{ij}]$

	1	2	3	4	5	6	7	8	9
1	×	[0]	6	5	9	6	6	5	7
2	4	×	[0]	4	10	6	4	6	9
3	[0]	3	×	5	10	7	3	0	7
4	5	7	9	×	[0]	0	3	7	8
5	4	8	10	[0]	×	10	6	9	7
6	4	4	8	6	11	×	[0]	4	5
7	3	5	4	0	2	9	×	[0]	6
8	0	6	3	5	8	6	9	×	[0]
9	7	3	7	9	6	[0]	5	5	×

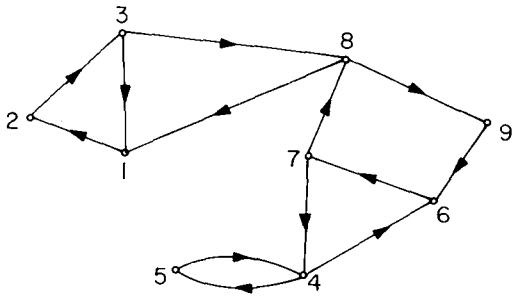

Fig. 2. G_0 after bounding procedure 1.

The new reduced cost matrix $[\bar{c}_{ij}]$ is shown in Table 4, and the corresponding admissible graph in Fig. 3.

Bounding procedure 3. Vertex 8 is an articulation point of the admissible graph of Fig. 3. The cutsets corresponding to this articulation point are $K_1' = (\{1, 2, 3\}, \{4, 5, 6, 7, 9\})$ and $K_1'' = (\{4, 5, 6, 7, 9\}, \{1, 2, 3\})$. Applying (20) to Table 4, we obtain $\nu_t = 2$, corresponding to element (5,1); and the lower bound becomes $B_3 = 40 + 2 = 42$. The new reduced cost matrix and the corresponding admissible graph are shown in Table 5 and Fig. 4 respectively.

Table 4

$[\bar{c}_{ij}]$

	1	2	3	4	5	6	7	8	9
1	×	[0]	6	5	7	6	6	5	7
2	4	×	[0]	4	8	6	4	6	9
3	[0]	3	×	5	8	7	3	0	7
4	5	7	9	×	[0]	0	3	7	8
5	2	6	8	[0]	×	8	4	7	5
6	4	4	8	6	9	×	[0]	4	5
7	3	5	4	0	0	9	×	[0]	6
8	0	6	3	5	6	6	9	×	[0]
9	7	3	7	9	4	[0]	5	5	×

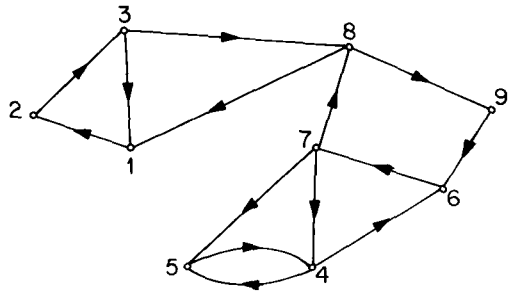

Fig. 3. G_0 after bounding procedure 2.

Table 5

$[c_{ij}]$ =

	1	2	3	4	5	6	7	8	9
1	×	[0]	6	3	5	4	4	5	5
2	4	×	[0]	2	6	4	2	6	7
3	[0]	3	×	3	6	5	1	0	5
4	3	5	7	×	[0]	0	3	7	8
5	0	4	6	[0]	×	8	4	7	5
6	2	2	6	6	9	×	[0]	4	5
7	1	3	2	0	0	9	×	[0]	6
8	0	6	3	5	6	6	9	×	[0]
9	5	1	5	9	4	[0]	5	5	×

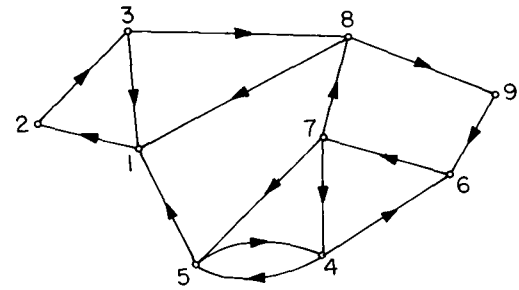

Fig. 4. G_0 after bounding procedure 3.

3. Finding a tour and improving the bound

Establishing whether a graph contains a tour (i.e., is hamiltonian) is, from the point of view of worst-case analysis, of the same order of difficulty as finding an optimal tour. However, for the vast majority of all possible graphs, the first problem is incomparably easier than the second one. We use a specialized implicit enumeration procedure, the multi-path method of [8, ch. 10], for finding a tour in G_0 if one can be found without exceeding a given time limit. Let $\hat{x}$ denote the solution associated with such a tour. If $\hat{x}$ satisfies with equality all inequalities associated with positive multipliers (i.e., if the tour defined by $\hat{x}$ crosses exactly once each cutset K_t, $t \in T_1$, contains exactly $|S_t| - 1$ arcs with

both ends in S_t for each S_t, $t \in T_2$, and contains exactly one arc of each pair of cutsets, K'_t, K''_t, $t \in T_3$), then $\hat{x}$ defines an optimal tour for the current subproblem, and the latter is fathomed.

For example, after bounding procedure 3, when G_0 is the graph of Fig. 4, the following tour is detected in G_0: $H = (1, 2, 3, 8, 9, 6, 7, 4, 5, 1)$. This tour satisfies with equality all four constraints with positive associated Lagrange multipliers; hence H is an optimal solution to the TSP and $B_3 = 42$ is its value.

For inequalities that are slack for $\hat{x}$, we attempt to strengthen the current lower bound by introducing some new inequalities (of the same type) that are tight for $\hat{x}$, and that admit positive multipliers. If the attempt is successful, it may also result in the removal of the inequality that is slack for $\hat{x}$.

3.1. *Bounding procedure 4*

Suppose the inequality (8a) associated with the cutset K_t is slack for $\hat{x}$, i.e., the tour $\hat{H}$ defined by $\hat{x}$ intersects K_t in more than one arc, and let $\hat{H} \cap K_t = \{(i_1, j_1), \ldots, (i_p, j_p)\}$. For every $(i_r, j_r) \in \hat{H} \cap K_t$, let S^r be a set of nodes containing j_r and such that, denoting $\bar{S}^r = N \setminus S^r$, the cutset $K_{tr} = (\bar{S}^r, S^r)$ contains no other arc of $\hat{H}$ than (i_r, j_r). Then the inequalities

$$\sum_{(i,j) \in K_{tr}} x_{ij} \geq 1, \quad r = 1, \ldots, p$$

are all satisfied with equality by $\hat{x}$. Since every K_{tr} contains an arc with zero reduced cost, namely the arc (i_r, j_r) also contained in K_t, the above inequalities do not admit a positive premium, unless the premium λ_t applied to K_t is reduced. If this is done, however, then a positive premium may be applicable to several of the sets K_{tr}, and the sum of these premia may well exceed the amount by which λ_t must be reduced, i.e., an improvement of the lower bound may be obtained. The conditions under which this is possible are stated in the next two propositions.

Proposition 6. *The tour $\hat{H}$ intersects the cutset $K_{tr} = (\bar{S}^r, S^r)$ only in the arc (i_r, j_r), if and only if the arcs of $\hat{H}$ with both ends in S^r form a path whose first node is j_r.*

Proof. Let $\hat{H} = \{i(1), \ldots, i(n)\}$, and without loss of generality, assume $(i_r, j_r) = [i(1), i(2)]$. Now suppose either $S^r = \{i(2)\}$, or the arcs of $\hat{H}$ with both ends in S^r form a path $\{i(2), \ldots, i(k)\}$. Then $\hat{H}$ intersects the cutset $K_{tr} = (\bar{S}^r, S^r)$ in the single arc $[i(1), i(2)] = (i_r, j_r)$.

Conversely, suppose $S^r \neq \{i(2)\}$ and the arcs of $\hat{H}$ with both ends in S^r either form a path P whose first node is not $i(2)$, or do not form a path. In the first case, $\hat{H} \cap K_{tr} = [i(h), i(h+1)]$, where $i(h+1)$ is the first node of P. In the second, the arcs of $\hat{H}$ with both ends in S^r form k paths $P_1, \ldots, P_k$, with $k \geq 2$;

and $\hat{H} \cap K_{tr} = [i(h_1), i(h_1+1)], \ldots, [i(h_k), i(h_k+1)]$, where $i(h_r+1)$ is the first node of P_r, $r = 1, \ldots, k$.

Proposition 7. *A positive premium can be applied to the cutset K_{tr} (provided that λ_t is decreased) if and only if*

$$(K_{tr} \smallsetminus K_t) \cap A_0 = \emptyset. \tag{22}$$

If $R \neq \emptyset$ is the set of those $r \in \{1, \ldots, p\}$ for which (22) holds, the maximum premium applicable to each K_{tr}, $r \in R$, is

$$\lambda^r = \min\left\{\lambda_t, \min_{(i,j) \in K_{tr} \smallsetminus K_t} \bar{c}_{ij}\right\} > 0; \tag{23}$$

provided the premium λ_t applied to K_t is replaced by

$$\hat{\lambda}_t = \lambda_t - \max_{r \in R} \lambda^r. \tag{24}$$

This replaces the current lower bound B by

$$B' = B + \sum_{r \in R} \lambda^r - \max_{r \in R} \lambda^r. \tag{25}$$

Proof. A decrease in λ_t increases the reduced costs of all arcs of K_t; hence makes it possible to apply a positive premium to the arcs of those, and only those, cutsets K_{tr} satisfying condition (22). The maximum size of the premium on K_{tr} is λ^r defined by (23), positive for those r for which (22) holds. The premia λ^r are however applicable only if λ_t is diminished by the amount of the largest premium applied to the arcs of any cutset K_{tr}, i.e., only if λ_t is replaced by $\hat{\lambda}_t$ of (24) (otherwise some of the reduced costs become negative). If this is done, the current lower bound B is replaced by

$$B' = B + \sum_{r \in R} \lambda^r + (\hat{\lambda}_t - \lambda_t)$$

which yields (25) after substituting for $\hat{\lambda}_t$.

Bounding procedure 4 looks for cutsets K_t to which a premium $\lambda_t > 0$ has been applied and which are intersected by the tour $\hat{H}$ in more than one arc. For each arc (i_r, j_r) of $\hat{H}$ that belongs to a cutset K_t, we try to find a set S^r of nodes containing j_r and satisfying the conditions of Propositions 6 and 7. As a matter of practicality, we first try $|S^r| = 2$, then $|S^r| = 3$ etc., until either we find a set which satisfies (22) or we find out that none exists. In the latter case, we take another arc of $\hat{H} \cap K_t$; in the former one, we compute λ^r, the premium to be applied to the cutset K_{tr}, and then take the next arc of $\hat{H} \cap K_t$. When all arcs of $\hat{H} \cap K_t$ have been examined, we compute the new value $\hat{\lambda}_t$ of the premium applicable to the arcs of K_t, and replace λ_t by $\hat{\lambda}_t$. All this replaces the reduced

costs $\bar{c}_{ij}$ by

$$\bar{c}'_{ij} = \begin{cases} \bar{c}_{ij} - \lambda^r, & (i,j) \in K_{tr} \setminus K_t, \quad r \in R, \\ \bar{c}_{ij} - \lambda^r + \max_{s \in R} \lambda^s, & (i,j) \in K_{tr} \cap K_t, \quad r \in R, \\ \bar{c}_{ij} + \max_{s \in R} \lambda^s, & (i,j) \in K_t \setminus (\bigcup_{s \in R} K_{ts}), \\ \bar{c}_{ij}, & \text{all other } (i,j) \in A, \end{cases}$$

and the lower bound B by B' defined in (25). If $\hat{\lambda}_t = 0$, i.e., $\max_{r \in R} \lambda^r = \lambda_t$, then the inequality corresponding to K_t vanishes from $L(w)$ and we have succeeded in replacing this constraint, slack for the solution $\hat{x}$, with a set of inequalities that are all tight for $\hat{x}$.

If T_1^+ is the index set of those inequalities (8a) that are slack for $\hat{x}$ and for which $|R| \neq \emptyset$, and if we attach a subscript t to the index sets R associated with each cutset K_t and to the premia λ^r indexed by R, then at the end of bounding procedure 4 we have the lower bound

$$B_4 = B_3 + \sum_{t \in T_1^+} \sum_{r \in R_t} \lambda_t^r - \sum_{t \in T_1^+} \max_{r \in R_t} \lambda_t^r. \tag{26}$$

Next we turn to the inequalities (8b).

3.2. Bounding procedure 5

Suppose the inequality (8b) defined by the node set S_t is slack for $\hat{x}$, and let G_t be the subgraph of G induced by the arc set $\hat{H} \cap (S_t, S_t)$. Note that $\hat{H} \cap (S_t, S_t)$ may be empty, since it is possible for a tour to contain all nodes in S_t without containing any arc with both ends in S_t; and when this is the case, no new inequalities can be derived from S_t.

Assume now that $\hat{H} \cap (S_t, S_t) \neq \emptyset$, and let $C^1, \ldots, C^s$ be the (connected) components of G_t. For $q \in Q = \{1, \ldots, s\}$, let S^q and A^q denote the node set and arc set, respectively, of C^q. By construction, each C^q is an open (directed) path, with $2 \leq |S^q| \leq |S_t| - 1$ and $|A^q| = |S^q| - 1$; hence $\hat{x}$ satisfies with equality each of the inequalities

$$-\sum_{i \in S^q} \sum_{j \in S^q} x_{ij} \geq 1 - |S^q|, \quad q \in Q. \tag{27}$$

Since $(S^q, S^q) \cap A_0 \neq \emptyset, \forall q \in Q$, these inequalities do not admit a positive penalty (without a change in the dual variables $\bar{u}$, $\bar{v}$), unless the penalty μ_t associated with S_t is reduced. If, however, this can be done, then each of the inequalities (27) admits a positive penalty and the current lower bound may be strengthened. The next proposition states the conditions for this.

Let F_t be the set of those arcs of G having both ends in S_t, but not both ends in the same set S^q, for any $q \in Q$; i.e., let

$$F_t = (S_t, S_t) - \bigcup_{q \in Q} (S^q, S^q).$$

Proposition 8. *A penalty $\mu_t^* > 0$ can be applied to each of the arc sets (S^q, S^q), $q \in Q$ (provided that the penalty μ_t is decreased), if and only if*

$$F_t \cap A_0 = \emptyset. \tag{28}$$

If (28) holds, then

$$\mu_t^* = \min_{(i,j)\in F_t} \bar{c}_{ij} > 0, \tag{29}$$

and the penalty μ_t^* can be applied to each arc set (S^q, S^q) if μ_t is replaced by $\mu_t - \mu_t^*$. This replaces the current bound B by

$$B' = B + (|Q| - 1)\mu_t^*. \tag{30}$$

Proof. Since $(S^q, S^q) \subset (S_t, S_t)$ and $(S^q, S^q) \cap A_0 \neq \emptyset$, $\forall q \in Q$, a penalty $\mu_t^* > 0$ can be applied to any (and all) of the arc sets (S^q, S^q) if and only if the penalty μ_t on the arc set (S_t, S_t) can be reduced by the same amount μ_t^*. This, however, is possible if and only if condition (28) is satisfied and μ_t^* does not exceed the reduced cost of any arc in F_t. When these conditions are present, all arc sets (S^q, S^q), $q \in Q$, can be penalized by the amount μ_t^* specified in (29), provided μ_t is replaced by $\mu_t - \mu_t^*$. The effect of all this on the lower bound is to add μ_t^* as many times as the number $|Q|$ of components of G_t, and to subtract μ_t^* once; which results in the bound B' of (30).

Bounding procedure 5 takes an inequality (8b) that is slack for $\hat{x}$, and forms the associated arc set F_t defined above. If (28) is not satisfied, we go to the next inequality that is slack for $\hat{x}$. If (28) holds, we calculate μ_t^* given by (29) and penalize by μ_t^* all arc sets (S^q, S^q), $q \in Q$, defined by the components of the graph G_t; while replacing the penalty μ_t on the arcs of (S_t, S_t), by $\mu_t - \mu_t^*$. This replaces the reduced costs $\bar{c}_{ij}$ by

$$\bar{c}'_{ij} = \begin{cases} \bar{c}_{ij} - \mu_t^*, & (i,j) \in F_t, \\ \bar{c}_{ij}, & (i,j) \in A \setminus F_t, \end{cases}$$

and the current lower bound B by B' defined by (30). If $\mu_t^* = \mu_t$, the inequality associated with S_t vanishes from $L(w)$, and we have succeeded in replacing this constraint, slack for $\hat{x}$, by a set of other constraints that are all tight for $\hat{x}$.

Next the procedure goes to another inequality (8b) that is slack for $\hat{x}$. When all such inequalities have been examined, let T_2^+ be the index set of those among them for which condition (28) was satisfied, and for each $t \in T_2^+$, let $|Q_t|$ be the number of components of the graph G_t. Bounding procedure 5 then produces the lower bound

$$B_5 = B_4 + \sum_{t\in T_2^+} (|Q_t| - 1)\mu_t^*.$$

Finally we turn to the inequalities (8c).

3.3. Bounding Procedure 6

Suppose the inequality (8c) associated with the articulation point k and the cutsets $K'_t = (S_t, \bar{S}_t \setminus \{k\})$, $K''_t = (\bar{S}_t \setminus \{k\}, S_t)$, is slack for $\hat{x}$, and let $\hat{H} \cap (K'_t \cup K''_t) = \{(i_1, j_1), \ldots, (i_p, j_p)\}$. For every $(i_r, j_r) \in \hat{H} \cap (K'_t \cup K''_t)$, $r = 1, \ldots, p$, we will specify a node set $S^r \subset N \setminus \{k\}$ such that, denoting $\bar{S}^r = N \setminus S^r$ and $K'_{tr} = (S^r, \bar{S}^r \setminus \{k\})$, $K''_{tr} = (\bar{S}^r \setminus \{k\}), S^r)$, the only arc of $\hat{H}$ contained in $K'_{tr} \cup K''_{tr}$ is (i_r, j_r).

Proposition 9. *The only arc of $\hat{H}$ contained in $K'_{tr} \cup K''_{tr}$ is (i_r, j_r), if and only if $S^r = S \setminus \{k\}$, where S is the node set of one of the two paths $P_1 = \{k, \ldots, i_r\}$ and $P_2 = \{j_r, \ldots, k\}$ in $\hat{H}$.*

Proof. Assume $S^r = S \setminus \{k\}$, where S is the node set of P_1 or P_2. In the first case, $\hat{H} \cap K'_{tr} = \{(i_r, j_r)\}$ and $\hat{H} \cap K''_{tr} = \emptyset$; in the second, $\hat{H} \cap K'_{tr} = \emptyset$ and $\hat{H} \cap K''_{tr} = \{(i_r, j_r)\}$. In both cases, (i_r, j_r) is the only arc of $\hat{H}$ contained in $K'_{tr} \cup K''_{tr}$.

Conversely, let (i_r, j_r) be the only arc of $\hat{H}$ contained in $K'_{tr} \cup K''_{tr}$. Then either $\{(i_r, j_r)\} = \hat{H} \cap K'_{tr}$ and $\hat{H} \cap K''_{tr} = \emptyset$, or $\{(i_r, j_r)\} = \hat{H} \cap K''_{tr}$ and $\hat{H} \cap K'_{tr} = \emptyset$. In the first case, $\hat{H}$ enters S^r from k rather than from some node of $\bar{S}^r \setminus \{k\}$, since $\hat{H} \cap K''_{tr} = \emptyset$; and it exits S^r exactly once, through i_r; hence $S^r = S \setminus \{k\}$, where S is the node set of P_1. In the second case, $\hat{H}$ exits S^r through an arc whose front end is k, rather than some node of $\bar{S}^r \setminus \{k\}$, since $\hat{H} \cap K'_{tr} = \emptyset$; and it enters S^r exactly once, through j_r; hence $S^r = S \setminus \{k\}$, where S is the node set of P_2.

Thus, if the node sets S^r, $r = 1, \ldots, p$, satisfy the conditions of Proposition 9, then the inequalities

$$\sum_{(i,j) \in K'_{tr} \cup K''_{tr}} x_{ij} \geq 1, \quad r = 1, \ldots, p$$

are all satisfied by $\hat{x}$ with equality. The next proposition states the conditions under which a positive premium can be applied to the sets $K'_{tr} \cup K''_{tr}$.

Proposition 10. *A positive premium can be applied to the arc set $K'_{tr} \cup K''_{tr}$ if and only if*

$$[(K'_{tr} \cup K''_{tr}) \setminus K_t] \cap A_0 = \emptyset. \tag{32}$$

If $R \neq \emptyset$ is the set of those $r \in \{1, \ldots, p\}$, for which (32) holds, the maximum premium applicable to each $K'_{tr} \cup K''_{tr}$, $r \in R$, is

$$\nu^r = \min\left\{\nu_t, \min_{(i,j) \in K} \bar{c}_{ij}\right\}, \tag{33}$$

where $K = (K'_{tr} \cup K''_{tr}) \setminus K_t$; provided the premium ν_t applied to K_t is replaced by

$$\hat{\nu}_t = \nu_t - \max_{r \in R} \nu^r. \tag{34}$$

This replaces the current lower bound B by

$$B' = B + \sum_{r \in R} \nu^r - \max_{r \in R} \nu^r. \tag{35}$$

Proof. Analogous to the proof of Proposition 7.

Bounding procedure 6 looks for indices $t \in T_3$ for which a positive premium ν_t has been applied to the arc set $K'_t \cup K''_t$, and for which $\hat{H} \cap (K'_t \cup K''_t) = \{(i_1, j_1), \ldots, (i_p, j_p)\}$, with $p \geq 2$. Given such a $t \in T_3$, for each $r \in \{1, \ldots, p\}$ we use the node set of the path $P_r = (k, \ldots, i_r)$ in $\hat{H}$, after removing from it node k, to derive an arc set of the form $K'_{tr} \cup K''_{tr}$ defined in Proposition 9. We then check whether $K'_{tr} \cup K''_{tr}$ satisfies (32), and if so, we calculate the premium ν^r to be applied to $K'_{tr} \cup K''_{tr}$; otherwise we move to the next $r \in \{1, \ldots, p\}$. When all arcs of $\hat{H} \cap (K'_t \cup K''_t)$ have been examined, we compute the value $\hat{\nu}_t$ of the premium applicable to the arcs of K_t, as given by (34), and replace ν_t by $\hat{\nu}_t$. All this replaces the reduced costs $\bar{c}_{ij}$ by

$$\bar{c}'_{ij} = \begin{cases} \bar{c}_{ij} - \nu^r & (i,j) \in (K'_{tr} \cup K''_{tr}) \setminus K_t, \quad r \in R \\ \bar{c}_{ij} - \nu^r + \max_{s \in R} \nu^s & (i,j) \in (K'_{tr} \cup K''_{tr}) \cap K_t, \quad r \in R \\ \bar{c}_{ij} + \max_{s \in R} \nu^s & (i,j) \in K_t \setminus \bigcup_{s \in R} (K'_{ts} \cup K''_{ts}) \\ \bar{c}_{ij} & \text{all other } (i,j) \in A \end{cases}$$

and the lower bound B by B' defined in (35).

As in the case of procedures 4 and 5, if $\hat{\nu}_t = 0$, the inequality associated with $K'_t \cup K''_t$ vanishes from $L(w)$ and is replaced by new inequalities that are tight for $\hat{x}$.

Let T_3^+ be the index set of those inequalities (8c) that are slack for $\hat{x}$ and for which $R \neq \emptyset$, and let us attach a subscript t to the index set R associated with $K'_t \cup K''_t$ and to the premia ν^r indexed by R. At the end of bounding procedure 6 we then have the lower bound

$$B_6 = B_5 + \sum_{t \in T_3^+} \sum_{r \in R_t} \nu_t^r - \sum_{t \in T_3^+} \max_{r \in R_t} \nu^r. \tag{36}$$

Naturally, if at any stage of the procedure the lower bound for the current subproblem matches the upper bound on v(TSP) given by the value of the best tour at hand, the current subproblem is fathomed.

At this point we may find ourselves in one of two possible situations: (α) we have found a tour in G_0, and used it to obtain the lower bound B_6 on the value of the current subproblem; or (β) the attempt to find a tour was unsuccessful, and B_3 is the best lower bound we have for the current subproblem. In case (β), we define $G_\epsilon = (N, A_\epsilon)$, with $A_\epsilon = \{(i,j) \in A \mid \bar{c}_{ij} \leq \epsilon\}$, where the $\bar{c}_{ij}$ are the current reduced costs and ϵ is the smallest number for which we are able to find a tour in G_ϵ within the given time limit. In either case, we denote by $\hat{H}$ the tour

at hand, by $\hat{x}$ the associated solution, by $\bar{c}_{ij}$, $(i, j) \in A$, the last set of reduced costs, and by B the lower bound for the current subproblem. Obviously $c\hat{x}$, where c is the original cost vector, is an upper bound on v(TSP), and the best such upper bound at each stage will be denoted by B^*.

3.4. Computational complexity of the bounding procedures

Each of the six bounding procedures discussed in Sections 2 and 3 is polynomially bounded. For each of them except for the first one, the number of operations required in the worst case is $O(n^3)$, where n is the number of cities. For procedure 1, this number is $O(n^4)$. Solving the assignment problem at the start also requires at most $O(n^3)$ operations.

At every node of the search tree, the bounding procedures are applied once (after solving the assignment problem, if necessary) in the order 1, 2, 3. If at that point the node was still not fathomed (i.e., the lower bound is still below the current upper bound), an attempt is made at finding a tour in G_0. Though there is no algorithm guaranteed to accomplish this in polynomial time, we let our implicit enumeration procedure run only for a fixed amount of time, that is an input parameter defined as a linear function of n. If a tour is found, bounding procedures 4, 5, 6 are applied in that order; otherwise we branch.

In conclusion, the amount of work performed at any given node of the search tree is $O(n^4)$ in the worst case.

4. Branching rules

Before branching, we attempt to fix some variables by using the bounds B and B^*. Let

$$Q_0 = \{(i, j) \in A \mid \bar{c}_{ij} \geq B^* - B\}.$$

It is not hard to see that, if the reduced costs $\bar{c}_{ij}$ are derived from the same dual solution and Lagrange multipliers as the lower bound B (as is the case here), then any solution x to TSP such that $cx < B^*$ must satisfy the condition $x_{ij} = 0$, $\forall (i, j) \in Q_0$. Hence we set $x_{ij} = 0$, $(i, j) \in Q_0$ for the current subproblem and its descendants, i.e., we replace A by $A \smallsetminus Q_0$.

Next we describe two branching rules, which we use intermittently. The first rule derives a disjunction from a conditional bound; the second rule derives one from a subtour-breaking inequality.

4.1. Disjunction from a conditional bound

A disjunction from a conditional bound can be obtained as follows [1]. Consider a family of sets $Q_k \subset A$, $k = 1, \ldots, p$, such that $\bar{c}_{ij} > 0$, $\forall (i, j) \in Q_k$, $k = 1, \ldots, p$. Then if the inequalities

$$\sum_{(i,j)\in Q_k} x_{ij} \geq 1, \quad k = 1, \ldots, p$$

were added to the constraint set of LP, the lower bound B could be improved by choosing appropriate multipliers for these inequalities. Further, if this improved bound (termed conditional, because of the hypothetical nature of the inequalities) matches the upper bound B^*, then every solution better than the one associated with B^* violates at least one of the above inequalities; i.e., satisfies the disjunction

$$\bigvee_{k=1}^{p} (x_{ij} = 0, \ \forall (i,j) \in Q_k). \tag{37}$$

To implement this principle, we first remove from $L(w)$ those inequalities (8a) and (8c) that are slack for $\hat{x}$ (the solution corresponding to the current tour) while the associated multiplier w_t is positive. Let $\hat{c}_{ij}$ denote the reduced costs, and $\hat{B}$ the lower bound, resulting from this removal. Next, we choose a minimum cardinality arc set $S \subset \hat{H}$, $S = \{(i_1, j_1), \ldots, (i_p, j_p)\}$, such that

$$\sum_{(i,j)\in S} \hat{c}_{ij} \geq B^* - \hat{B}, \tag{38}$$

and construct a $p \times |A|$ 0–1 matrix $D = (d_{ij}^k)$ with as many 1's in each column as possible, subject to the conditions $d_{i_k j_k}^k = 1$, $k = 1, \ldots, p$, and

$$\sum_{k=1}^{p} d_{ij}^k \, \hat{c}_{i_k j_k} \leq \hat{c}_{ij}, \quad (i,j) \in A.$$

It can then be shown (see [1]) that every solution x to TSP such that $cx < B^*$ satisfies the disjunction (37) for the sets $Q_k = \{(i,j) \in A \mid d_{ij}^k = 1\}$, $k = 1, \ldots, p$.

The disjunction (37) creates p subproblems. In the k-th subproblem we have $x_{ij} = 0$, $(i,j) \in Q_k$, and since $(i_k, j_k) \in \hat{H} \cap Q_k$, the tour $\hat{H}$ becomes infeasible for each of the subproblems. On the other hand, the current solution to AP remains feasible for each of the subproblems.

4.2. *Disjunction from a subtour breaking inequality*

A disjunction from a subtour breaking inequality is obtained in the usual way; i.e., if S is the arc set of a subtour of the AP solution, then every solution to TSP satisfies the disjunction:

$$\bigvee_{(i_k, j_k)\in S} (x_{i_k j_k} = 0 \text{ and } x_{i_l j_l} = 1, \forall l \leq k-1) \tag{39}$$

At an arbitrary node of the branch and bound tree, a subset S' of the arc set S (of the subtour selected for branching) may already have been fixed to be in the solution. In this case set S in disjunction (39) is replaced by $S \smallsetminus S'$. Branching on (39) creates $|S \smallsetminus S'|$ subproblems. For each of these subproblems, the AP solution to the parent problem becomes infeasible.

In choosing the arc set S for the disjunction (39), it is desirable to give preference to subtours (of the current AP solution) having either a minimum number of arcs (min $|S|$), or a minimum number of free arcs (min $|S \setminus S'|$). In the computational tests discussed in the next section we used the first of these two criteria.

As to the two disjunctions (37) and (39), an efficient procedure must use them intermittently, since (37) can on occasion be considerably stronger than (39), while at other times it can be much weaker. We tried several rules for mixing them, and the one actually used in the tests is discussed in the next section.

5. Implementation and computational experience

Our algorithm was programmed in FORTRAN IV for the CDC 7600 and tested on a set of 120 randomly generated asymmetric TSP's of sizes varying between 50 and 325 cities. Here we discuss some features of the implementation, give the computational results, and interpret them.

5.1. Use of sparsity

Unlike in the case of those symmetric TSP's whose costs are based on distances and can therefore be generated whenever needed from the $2n$ coordinates of the cities, in the case of the asymmetric TSP one has to explicitly store the costs, whose number in case of a complete graph is $n(n-1)$. However, as discussed at the beginning of Section 4, our procedure uses the bounds to fix at 0 certain variables, and the number of variables for which this can be done before the first branching is usually very high. Therefore at that point we actually remove from the graph all those arcs whose variables can be fixed at 0, and from then on we work with a graph (usually quite sparse) represented by a list of nodes and a list of arcs with their costs. Additional fixing of variables (at 0 or 1) later in the procedure is handled via cost modifications.

5.2. Branching and node selection

The two types of branching discussed in Section 4 are used intermittently according to the following rule. A branching of type 1 (based on disjunction (37)) is performed whenever a set of arcs $S \subset \hat{H}$ can be found, such that (i) inequality (38) is satisfied; (ii) $|S| < (r/2)+1$, where r is the number of arcs in the smallest subtour in the current AP solution; and (iii) at least $n/3$ variables can be fixed at 0 on each branch.

Whenever any of the above conditions is violated, a branching of type 2 (using disjunction (39)) is performed.

The node selection rule used in the code is to choose a successor of the current node whenever available, and otherwise to select a node k for which the

following evaluation attains its minimum:

$$E(k) = [B(k) - v(\mathrm{AP})] \cdot \frac{s(k)-1}{s(0)-s(k)}.$$

Here $B(k)$ is the lower bound for subproblem k, $v(\mathrm{AP})$ is the value of the (initial) AP, while $s(0)$ and $s(k)$ are the number of subtours in the solutions to the initial AP and the current one (at node k), respectively. The integer $s(k)$ is used as a measure of the "distance" of the AP solution at node k from an optimal tour.

5.3. *Information stored for each subproblem*

All subproblems are stored on a linked list in order of increasing lower bounds. For each subproblem k the following information is stored: the AP solution; the value of the associated bound; a pointer to the father node of node k; a code to indicate the type of branching (one of the 2 types described above) that produced node k; the number of sons of the father of node k; the rank (index) of node k among its brothers; a list of the arcs of S in (38) (if node k was obtained from a disjunction (37)), or a pointer to the subtour in the AP solution corresponding to S in (39), (if node k was obtained from a disjunction (39)), and, finally, a list of the operations (in coded and ordered form) which produced the current matrix $[c_{ij}]$ from the matrix for the predecessor node.

5.4. *Computational results*

The above described code was run on the CDC 7600 to solve 120 randomly generated test problems whose associated (directed) graphs are complete and whose cost coefficients were drawn from a uniform distribution of the integers in the range [1, 1000]. The problems belong to 12 classes based on size, with $n = 50, 75, \ldots, 300, 325$, and with 10 problems in each class. Table 6 summarizes the results. These results are quite remarkable, in that the number of nodes generated is surprisingly small, and seems to increase only slightly faster than the number of cities. This is also illustrated on Fig. 5, where the slope of the curve is only slightly steeper for $200 \leq n \leq 325$ than for $50 \leq n \leq 200$. Note, also, that the maximum time required to solve any one of the 120 problems was 82 seconds.

Table 7 shows the ratio between computing time and the number of variables, whose growth with problem size is surprisingly slow.

Since the average cost of the various bounding procedures is not proportional to their usefulness, we have tested each of the procedures individually and in subsets to see whether their use pays off. The outcome of our tests was that using all 6 bounding procedures is more efficient than using any subset of these procedures.

An interesting feature of the approach discussed here is the large number of

Table 6
Computational results on random asymmetric TSP's

Class	n	Average no. of nodes	Computing time (CDC 7600 seconds)		Percentage of nodes fathomed by						Percentage of time spent on						
			Average	Maximum	B_0	B_1	B_2	B_3	H	$B_{4,5,6}$	B_0	B_1	B_2	B_3	H	$B_{4,5,6}$	Other
1	50	12.3	.20	.88	25.2	29.1	6.9	10.4	21.4	7.0	5.3	15.5	26.0	15.3	1.2	20.1	16.6
2	75	26.6	.29	.93	26.7	26.6	10.3	10.1	19.8	6.5	5.3	15.4	26.9	12.6	0.8	20.7	18.3
3	100	39.1	.71	1.41	21.8	30.7	9.5	14.9	16.5	6.6	4.9	15.1	27.8	14.1	0.8	23.1	15.2
4	125	42.7	1.13	2.07	19.6	34.4	12.0	13.8	16.5	3.7	4.8	16.0	23.9	17.3	1.1	26.1	10.8
5	150	45.7	1.97	3.30	19.6	28.5	15.1	12.9	13.3	10.6	3.9	16.3	24.8	14.9	1.3	26.6	12.2
6	175	58.3	4.18	6.68	20.1	28.7	17.4	13.7	15.4	4.7	4.6	16.9	28.4	14.8	0.9	19.3	15.1
7	200	63.4	6.06	19.33	14.7	33.9	14.9	17.2	12.1	7.6	4.1	16.5	29.7	16.7	1.6	19.1	12.3
8	225	84.1	10.44	18.65	11.1	29.6	22.6	16.8	8.8	11.1	3.8	16.3	29.1	16.4	1.8	20.1	12.5
9	250	88.5	13.65	17.43	12.5	29.7	17.2	21.9	9.2	9.5	3.3	17.6	27.7	17.3	1.6	20.2	12.3
10	275	106.4	21.74	68.86	9.3	25.4	20.5	19.3	8.5	7.0	2.9	18.5	27.9	16.5	1.9	18.9	13.4
11	300	124.1	38.37	55.15	10.7	28.9	19.1	23.8	6.9	10.6	3.1	18.9	29.1	14.1	2.1	20.0	12.7
12	325	141.8	49.66	81.57	7.7	32.3	18.7	21.6	7.8	11.9	2.6	20.1	30.3	16.0	2.1	17.1	11.8

Notes: n: number of cities; B_0: lower bound obtained by solving the current AP and adding to $v(\text{AP})$ a penalty derived from $[\bar{c}_{ij}]$; B_1, B_2, B_3: lower bound obtained by procedures 1, 2 and 3, respectively: H: upper bound obtained by finding a tour in G_0 satisfying the condition of Proposition 3; $B_{4,5,6}$: lower bound obtained by procedures 4 to 6; Other: branching, node selection, updating, etc.

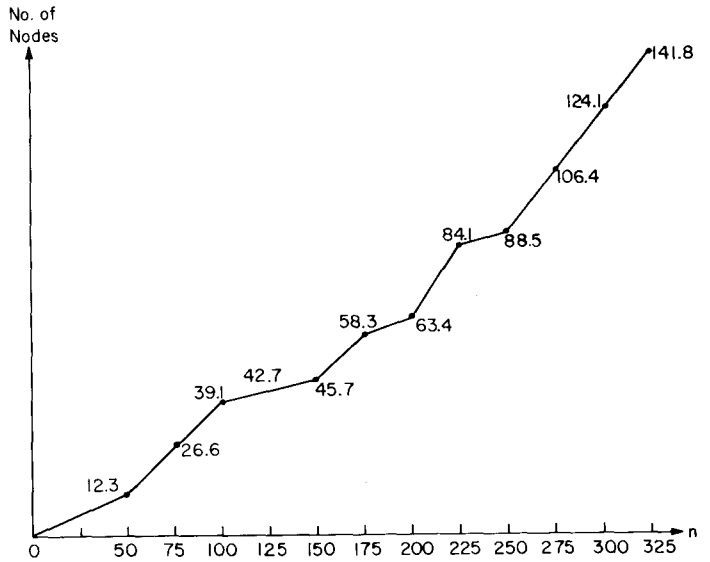

Fig. 5. Number of nodes in the search tree as a function of number of cities (n).

Table 7
Ratio of computing time to number of variables

n	50	75	100	125	150	175	200	225	250	275	300	325
$\frac{T \times 10^5}{n^2}$	8	5	7	7	8	13	15	20	21	28	42	47

T = computing time in seconds; n = number of cities.

arcs that can be removed from the graph at the root node of the search tree, as a result of the test discussed at the beginning of Section 4 (see Table 8). This shows the power of the bounding procedures used in our approach. For a comparison, if only the bound obtained from AP is used, then the percentage of variables removed in problem classes 1, 2 and 3 is on the average 87% (and this percentage does not seem to increase with problem size).

Table 9 compares our computational results with those reported for two of the most successful other codes available for asymmetric TSP's. The Smith–Thompson [18] code is an implementation of a branch and bound algorithm that generates bounds from the AP solution by estimating the effect of fixing variables, and branches on the arcs of a subtour. It is the most successful of the 3 versions evaluated in [18]. The Kubo–Okino [16] code is also an AP-based branch and bound procedure. Both codes were tested by their authors on a set of randomly generated asymmetric TSP's with integer coefficients drawn from a uniform distribution over the interval [1, 1000], just like our problems. Thus,

Table 8
Percent of arcs removed on the average at the root node

Problem class	1	2	3	4	5	6	7	8	9	10	11	12
$\frac{A \times 100}{\Sigma}$	95.3	96.4	97.1	97.3	97.5	97.6	97.9	98.1	98.4	98.3	98.6	98.7

A = Arcs removed (average); Σ = Total arcs.

Table 9
Comparison of average running times (seconds)

n	Smith–Thompson [18] UNIVAC 1108 5 problems in each class	Kubo–Okino [16] HITAC 8800 10 problems in each class	Balas–Christofides CDC 7600 10 problems in each class
50	1.7	4.8	.20
60	9.3	–	–
70	8.5	–	–
75	–	–	.29
80	13.8	–	–
90	42.0	–	–
100	53.0	45.2	.71
110	22.3	–	–
120	62.9	–	–
125	–	–	1.13
130	110.1	–	–
140	165.2	–	–
150	65.3	109.5	1.97
160	108.5	–	–
170	169.8	–	–
175	–	–	4.18
180	441.4	–	–
200	–	$>281.3^a$	6.06
225	–	–	10.44
250	–	$>252.5^b$	13.65
275	–	–	21.74
300	–	$>275.0^c$	38.37
325	–	–	49.66

[a] average for 7 problems
[b] average for 8 problems
[c] average for 6 problems
} remaining problems could not be solved in 600 seconds each

though the three codes were run on different sets of problems, the results should be comparable after correction for differences in computer speed. The CDC 7600 is about 3 times as fast as the UNIVAC 1108, and the HITAC 8800 is comparable in speed to the UNIVAC 1108 [19]. The largest problems solved by Smith and Thompson had 180 cities. Kubo and Okino went up to 300 cities, but beyond 150 cities were unable to solve all 10 problems within the time limit of

600 seconds for each problem. The most relevant aspect of the comparison, which is unaffected by computer speed, is the change in the ratio min{S–T, K–O}/(B–C) as a function of problem size. Here S–T, K–O and B–C stands for the computing times reported for three codes. While for the 50 city problems this ratio is about 8/1, for the 100 city problems it becomes about 55/1, and for the 180–200 city problems it is of the order of 100/1.

6. The symmetric case

Our algorithm can of course be applied to symmetric TSP's as it is, but it would not be efficient for such problems in its present form. This is so because of the well known fact that AP's associated with asymmetric TSP's tend to have optimal solutions involving a large number of subtours of length two. However, our approach can be adapted to the symmetric case by replacing the assignment problem with the 2-matching problem as the basic relaxation of the TSP. We are in the process of developing such an algorithm for the symmetric TSP.

References

[1] E. Balas, "Cutting planes from conditional bounds: A new approach to set covering", *Mathematical Programming Studies* 12 (1980) 19–36.
[2] E. Balas and N. Christofides, "A new penalty method for the traveling salesman problem", Paper presented at the Ninth International Symposium on Mathematical Programming, Budapest, August 1976.
[3] M. Bellmore and J.C. Malone, "Pathology of traveling salesman subtour elimination algorithms", *Operations Research* 19 (1971) 278–307.
[4] M. Bellmore and G.L. Nemhauser, "The traveling salesman problem: A survey", *Operations Research* 16 (1968) 538–558.
[5] R.E. Burkard, "Traveling salesman and assignment problems: A survey", *Annals of Discrete Mathematics* 4 (1979) 193–217.
[6] N. Christofides, "The shortest hamiltonian chain of a graph", *SIAM Journal of Applied Mathematics* 19 (1970) 689–696.
[7] N. Christofides, "Bounds for the traveling salesman problem", *Operations Research* 20 (1972) 1044–1055.
[8] N. Christofides, *Graph theory: An algorithmic approach* (Academic Press, New York, 1976).
[9] N. Christofides, "The traveling salesman problem", in: N. Christofides, A. Mingozzi, P. Toth and C. Sandi, *Combinatorial Optimization* (Wiley, New York, 1979) pp. 131–150.
[10] G.B. Dantzig, D.R. Fulkerson and S. Johnson, "Solution of a large scale traveling salesman problem", *Operations Research* 2 (1954) 393–410.
[11] W.L. Eastman, "Linear programming with pattern constraints", Unpublished Ph.D. Dissertation, Harvard University, 1958.
[12] K.H. Hansen and J. Krarup, "Improvements of the Held–Karp algorithm for the symmetric traveling salesman problem", *Mathematical Programming* 7 (1974) 87–96.
[13] M. Held and R. Karp, "The traveling salesman problem and minimum spanning trees", *Operations Research* 18 (1970) 1138–1162.
[14] M. Held and R. Karp, "The traveling salesman problem and minimum spanning trees, II", *Mathematical Programming* 1 (1971) 6–25.

[15] V. Klee, "Combinatorial optimization: What is the state of the art", *Mathematics of Operations Research* 5 (1980) 1–26.
[16] H. Kubo and N. Okino, "A new practical solution for large traveling salesman problems", *Journal of the Operations Research Society of Japan* 19 (1976) 272–285 [In Japanese.]
[17] D.M. Shapiro, "Algorithms for the solution of the optimal cost and bottleneck traveling salesman problem", Unpublished Sc.D. Thesis, Washington University, St. Louis, 1966.
[18] T.C.H. Smith and G.L. Thompson, "Computational performance of three subtour elimination algorithms for asymmetric traveling salesman problems", *Annals of Discrete Mathematics* 1 (1977) 495–506.
[19] *Computer Review*, GML Corp., Lexington, MA (1979).